VILNIAUS UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS FAKULTETAS
KOMPIUTERIJOS KATEDRA

Baigiamasis bakalauro darbas

Rašytų skaitmenų atpažinimas taikant dirbtinius neuroninius tinklus

Atliko: 4 kurso, 4 grupės studentas
Laurynas Brazauskas

Darbo vadovas:
docentas Algirdas Bastys

Vilnius
2009

Turinys

Anotacija

Darbo esmė buvo ištirti dirbtinius neuroninius tinklus, kaip jie veikia ir kaip yra taikomi klasifikavimo uždaviniams spręsti. Vienas iš darbo tikslų buvo išanalizuoti, kaip dirbtinius neuroninius tinklus galima pritaikyti sprendžiant pasirinktą klasifikavimo uždavinį – rašytų skaitmenų atpažinimą. Kitas tikslas buvo sukurti programą, kuri atpažintų ranka rašytus skaitmenis. Užbrėžti tikslai buvo įvykdyti – išnagrinėti dirbtinių neuroninių tinklų pagrindai ir parašyta programa, kuri atpažįsta pateikiamus skaitmenis apie 91 procento tikslumu. Programa buvo palyginta su kitų autorių darbais.

Handwritten digit recognition using artificial neural networks

The essence of work was to investigate the field of artificial neural networks, how they work and how they can be used in solving classification problems. One of the goals of work was to analyze how artificial neural networks could be used to solve the chosen classification problem – recognizing handwritten digits. Another goal of work was to use the investigation results in creating a software program that would recognize handwritten digits.

During the course of work a lot of information was researched regarding artificial neural networks, image feature extraction, classification and handwritten digit recognition. To get the program working some code was written that loads the images from the MNIST digit database [LC98]. To feed the images to the neural network some image feature extraction code was written. Furthermore some investigations and experiments were done to decide what architecture of the neural network should be used.

In conclusion, the set goals were accomplished. The core of artificial neural networks was analyzed and a software program was written that recognizes handwritten digits with about 91 percent accuracy. The accuracy is quite good, though compared to what others have accomplished, it is a bit behind. It was also shown that artificial neural networks are very applicable in solving such problems were an algorithmic approach is not so well suited. A future improvement of the program could be to use more advanced techniques such as convolutional neural networks which currently reach highest accuracies (above 99%). The created program uses a multilayer perceptron which is discussed in detail in this work.

Įvadas

Darbo tema buvo dirbtiniai neuroniniai tinklai (DNT) ir jų pritaikymas klasifikavimo uždavinių sprendimuose. Nagrinėjimui pasirinktas konkretus klasifikavimo uždavinys – rašytų skaitmenų atpažinimas.

Dirbtiniai neuroniniai tinklai yra taikomi įvairiose realaus pasaulio srityse. Pastaruoju metu ši technologija jau išlindo iš eksperimentavimo laboratorijų ir yra naudojama aktualiems uždaviniams spręsti. Keletas pavyzdžių būtų tokie: automobilių numerių, žmonių veidų atpažinimas, ekonominės prognozės, kredito išdavimo rizikos skaičiavimas ir taip toliau. Dėl DNT pagrįstumo ant biologinės neuroninės sistemos, jie yra gana perspektyvūs sprendžiant problemas, kurias žmonės atlieka be didelių pastangų, tokias kaip klasifikavimas ir prognozavimas. Viena iš tokių problemų yra rašytų skaitmenų atpažinimas, kuris yra plačiai taikomas pašto kodų ir banko čekių nuskaityme. Šio proceso automatizavimas padeda greitai ir efektyviai rūšiuoti laiškus ir čekius.

Idėja, kad kompiuteris gali spręsti labai sudėtingus uždavinius nenaudodamas programuotojo aprašytų sudėtingų algoritmų, yra labai išsiskirianti iš kitų uždavinių sprendimų idėjų, pagrįstų griežta logika ir tiesioginiu priėjimu prie problemos. DNT veikimas remiasi daugiau duomenimis, o ne logika. Parašyta programa tai patvirtina.

Uždavinys ir rezultatai

Šiame darbe sprendžiamas uždavinys buvo rašytų skaitmenų atpažinimas. Kadangi pasirinktas sprendimo būdas buvo DNT taikymas, tai pirmiausia reikėjo išanalizuoti jų teorinį pagrindą. Nemažai buvo apžvelgta informacijos apie DNT ir jų veikimo principus. Taip pat buvo nagrinėjami taikymai skirtingose srityje (prognozavimo, klasifikavimo, ir pan.) ir su tuo susijusios problemos, tokios kaip tinkamo tinklo išdėstymo ir apmokymo strategijos parinkimas.

Kad efektyviau pritaikyti principus, tinkančius darbe sprendžiamam uždaviniui, buvo išnagrinėtos kitų žmonių sukurtos sistemos ir programos. Buvo nagrinėta viešai pateikiama rašytų skaitmenų atpažinimo programa [One06]. Taip pat buvo peržvelgti kitų žmonių darbai rašytų skaitmenų atpažinimo srityje – vienas baigiamasis darbas [Wu03] ir keli straipsniai internete.

Kad teorija neliktų tik teorija, buvo parašyta programa, atpažįstanti rašytus skaitmenis dirbtinių neuroninių tinklų dėka. Galutinė programa atpažįsta skaitmenis apie 91 procento

tikslumu. Tai gana geras rezultatas, nors ir atsilieka nuo kitų žmonių pasiekimų (virš 99 procentų tikslumas). Darbas parodo, kad DNT yra puikiai pritaikomi klasifikavimo uždavinių sprendimuose. Skaitmenų paveiksliukai buvo naudojami iš MNIST duomenų bazės [LC98], kurioje yra 60000 rašytų skaitmenų skirtų DNT apmokymui ir 10000 rašytų skaitmenų skirtų DNT patikrinimui (testavimui). Tokie dideli pavyzdinių duomenų kiekiai patvirtina programos veikimo tvirtumą.

1. Dirbtiniai neuroniniai tinklai

Dirbtiniai neuroniniai tinklai buvo pradėti nagrinėti dar XX amžiaus viduryje, tačiau tik apie 1980 metus jie tapo pakankamai išvystyti, kad būtų taikomi rimtesniems uždaviniams spręsti. Per keletą pastarųjų metų DNT sulaukė ypatingai daug susidomėjimo dėl jų pritaikymo įvairiose srityse, tokiose kaip fizika, inžinerija, finansai, medicina ir panašiai. DNT yra ypač sėkmingai naudojami prognozavimo, klasifikavimo ir kontrolės problemų sprendimuose. Pagrindinės DNT savybės nulemiančios jų tokį populiarumą yra šios:

- *Gebėjimas spręsti sudėtingas problemas*. DNT sugeba modeliuoti ypatingai sudėtingas funkcijas. Taip pat, DNT yra netiesinis ir gali spręsti uždavinius turinčius labai daug kintamųjų.
- *Naudojimo paprastumas*. DNT mokosi iš pavyzdžių. Tam, kad sėkmingai naudoti DNT, užtenka pasirinkti tinkamą jo struktūrą ir tinkamus duomenis jam apmokyti. Mokydamasis iš šių duomenų, DNT susitvarko savo vidinę struktūra taip, kad ši taptų naudotina pateiktai problemai spręsti.
- *Atsparumas triukšmui*. Tinkamai apmokytas DNT sugeba sėkmingai spręsti uždavinius su triukšmingais duomenimis. Iš pateiktos įvesties jis sugeba ištraukti esminę informaciją, reikalingą uždavinio sprendimui.

DNT yra pritaikomi praktiškai bet kokioje situacijoje, kur egzistuoja ryšys tarp žinomų kintamųjų arba įvesties ir ieškomų kintamųjų arba išvesties. Jei yra toks ryšys, kad ir labai sudėtingas, tai tinkamai parinktas DNT būtinai jį ras.

1.1. Struktūra

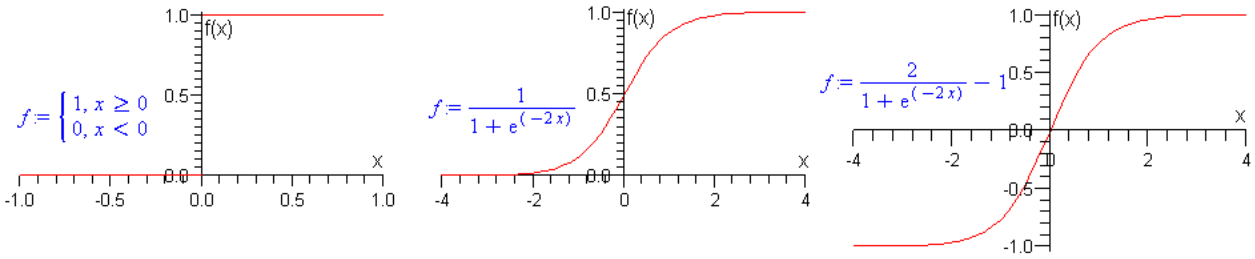
Dirbtiniai neuroniniai tinklai yra apytiksliai elektroniniai modeliai pagrįsti smegenų neuronų struktūra. DNT susideda iš labai paprastų ir masiškai sujungtų celių, vadinamų dirbtiniais neuronais. Dirbtinis neuronas turi keturias pagrindines sudedamąsias dalis. Šios yra tokios:

1. *Įvestis ir svoriai*. Neuronas gauna vieną ar daugiau įvesčių iš pradinių duomenų arba iš kitų to pačio tinklo neuronų išvesčių. Kiekviena neurono įvestis ateina per jungtį, kuri turi svorį (angl. weight) ar stiprumą (angl. strength). Neuronas taip pat turi vieną nepriklausomą (vidinę) įvestį – slenkstį (angl. threshold) ar poslinkį (angl. bias).
2. *Sumavimo santaka*. Visos neurono įvesčių reikšmės yra pasveriamos atitinkamais svoriais ir tada sudedamos. Prie šios sumos taip pat pridedama ir slenksčio reikšmė. Tai

yra standartinė ir plačiausiai naudojama sumavimo formulė, tačiau įmanoma naudoti ir kitokius operatorius (pvz.: reikšmių vidurkis, didžiausia ar mažiausia reikšmė, logiškai sudėtos reikšmės su AND, OR, XOR ir pan.) [AM92].

3. *Aktyvavimo funkcija*. Gauta suma toliau neurone yra perleidžiama per aktyvavimo funkciją, kuri tiesiog transformuoja šią sumą (x) į galutinį rezultatą (f(x)). Funkcijos algoritmas parenkamas priklausomai nuo sprendžiamos problemos. Kelios iš populiariausių funkcijų yra šios (žiūrėti 1 pav.):

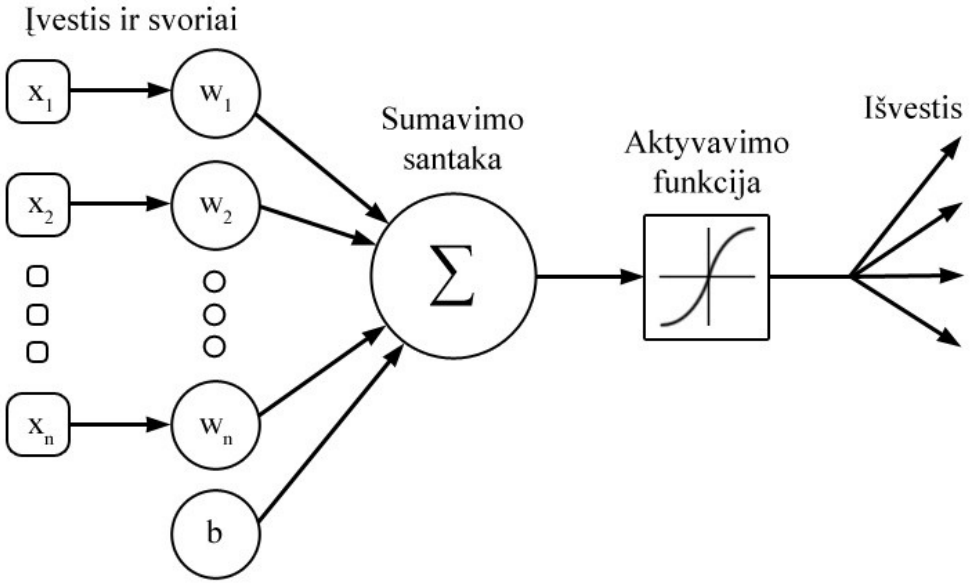
   - Slenksčio funkcija (angl. threshold function) – gražina 0, jei argumentas mažesnis už tam tikrą slenkstį (dažniausiai 0), kitu atveju gražina 1.
   - Sigmoidinė funkcija (angl. sigmoid function) – funkcijos grafikas yra S raidės tipo, kur reikšmės priklauso intervalui [0; 1].
   - Dvipolė sigmoidinė funkcija (angl. bipolar sigmoid function) – panaši kaip sigmoidinė, tik reikšmės kinta intervale [-1; 1], dėl ko ir yra vadinama dvipole, nes yra simetriška Y = 0 ašiai.



1 pav. Slenksčio (kairėje), sigmoidinė (viduryje) ir dvipolė sigmoidinė (dešinėje) funkcijos.

4. *Išvestis*. Neuronas gautą aktyvavimo funkcijos rezultatą siunčia prijungtiems neuronams arba pateikia jį kaip galutinį DNT rezultatą.

Bendras dirbtinio neurono vaizdas parodytas toliau (2 pav.).



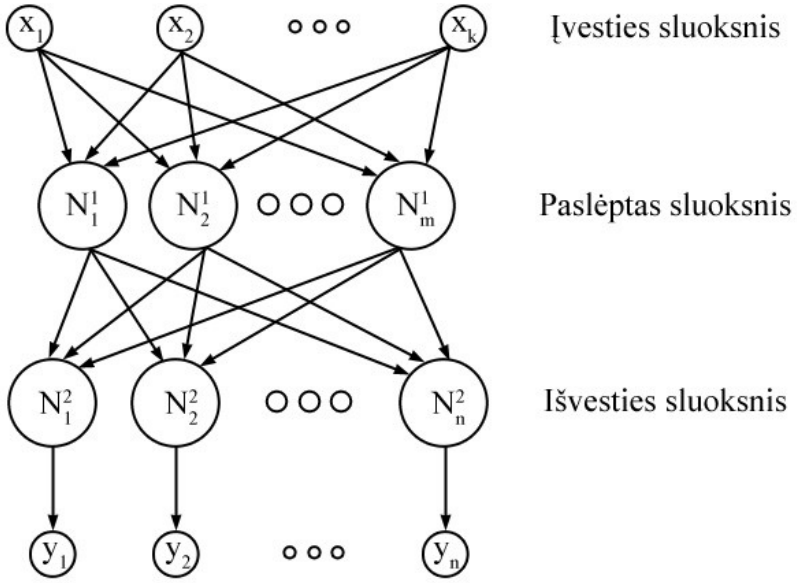
2 pav. Dirbtinio neurono sandara. x_i – i-oji įvesties reikšmė, w_i – i-asis svoris x_i pasverti, b – slenkstis.

1.2. Daugiasluoksniai perceptronai

Kai be apribojimų kalbama apie DNT dažniausiai turima omenyje daugiasluoksnius perceptronus (DSP) (angl. Multilayer perceptron (MLP)). Taip yra dėl to, kad DSP yra plačiausiai naudojami ir labiausiai išnagrinėti. Taip pat DSP yra vieni iš lengviausiai suprantamų neuroninių tinklų. Žinoma, yra daug ir kitokių tipų neuroninių tinklų: tikimybiniai, bendros regresijos, radialinio pagrindo funkcijų, Kohoneno, pasikartojantys ir dar daug kitų.

DSP yra sluoksniuotas vienkryptis (angl. feedforward) neuroninis tinklas. Šis tinklas viena kryptimi sujungia neuronus per sluoksnius. Kiekvienas neuronas priklauso kokiam nors sluoksniui ir jungiasi tik su neuronais, kurie yra iš kito sluoksnio. Kiekvienas sluoksnis yra pilnai sujungtas su po jo einančiu sluoksniu. Standartiškai DSP susideda iš tokių sluoksnių:

- *Įvesties sluoksnis*. Šis sluoksnis vienintelis neturi savyje neuronų, tačiau pasirūpina pradinių duomenų perdavimu pirmajam paslėptam sluoksniui. Įvestis parenkama pagal turimus duomenis problemos sprendimui.
- *Paslėpti sluoksniai*. Šių sluoksnių dažniausiai būna vienas arba du, priklausomai nuo sprendžiamo uždavinio sudėtingumo. Neuronai šiuose sluoksniuose modeliuoja įvesties duomenų šablonus. Be šių sluoksnių DNT sugeba spręsti tik tiesinius uždavinius. Pvz.: NOT, AND ir OR loginės operacijos. XOR operacija yra netiesinė (jos reikšmių sričių negalima atskirti tiese) ir dėl to jai spręsti yra būtinas bent vienas paslėptas sluoksnis.
- *Išvesties sluoksnis*. Šio sluoksnio neuronų išvestys sudaro sprendžiamo uždavinio galutinį rezultatą.



3 pav. Daugiasluoksnio perceptrono modelis su vienu paslėptu sluoksniu. N^i_j – i-ojo sluoksnio j-asis neuronas, x_i – i-oji įvestis į DSP, y_i – i-oji išvestis iš DSP.

Daugiasluoksniai perceptronai dėl savo unikalios architektūros sugeba modeliuoti labai sudėtingas funkcijas, kur jų sudėtingumas apsprendžiamas sluoksnių kiekiu ir neuronų kiekiu kiekviename sluoksnyje. Su vienu ar dviem paslėptais sluoksniais, jie sugeba surasti tikslų įvesties-išvesties planą. Tai jie daro vien tik keisdami neuronų įvesčių svorius ir slenksčius. Kokiu būdu šie kintamieji yra modifikuojami bus paaiškinta kitame skyrelyje.

1.2.1. Klaidos sklidimo atgal algoritmas

DSP yra stebimas (angl. supervised) neuroninis tinklas. Tai reiškia, kad šis tinklas mokosi su mokytojo pagalba, kur mokytojas pasakinėja teisingus atsakymus, kad tinklas galėtų pasitaisyti. DSP apmokymui yra naudojamas klaidos sklidimo atgal (angl. error backpropagation) algoritmas. Tai vienas iš paprasčiausių ir bendriausių stebimo mokymo metodų. Kiti metodai gali būti greitesni ar turėti kitų norimų savybių, tačiau retas yra aiškesnis ir lengviau suprantamas.

Klaidos sklidimo algoritmą galima aprašyti taip:

1. Imamas dar neapmokytas DSP. Šio tinklo neuronų svoriai ir slenksčiai yra inicializuojami atsitiktinėmis reikšmėmis.
2. Vienas įvesties šablonas (angl. input pattern) perleidžiamas per tinklą ir gaunamas išvesties šablonas (angl. output pattern). Kiekvienas neuronas atsimena savo išvesties reikšmę, kuri bus reikalinga klaidos skaičiavime.
3. Judama per sluoksnius nuo paskutinio iki pirmo ir kiekvienam neuronui paskaičiuojama

jo klaida. Neuronui klaida yra skaičiuojama per kelis žingsnius:

I. Jei tai neuronas iš paskutinio sluoksnio: iš trokštamos išvesties reikšmės (pateikiama mokytojo) atimama gauta neurono išvesties reikšmė.

II. Jei tai neuronas ne iš paskutinio sluoksnio: sumuojamos kito sluoksnio neuronų jau paskaičiuotos klaidos padaugintos iš svorių, kurie jungiasi iš dabartinio neurono.

III. Po to šis rezultatas yra padauginama iš neurono aktyvavimo funkcijos išvestinės reikšmės neurono įvesties reikšmei. Taip gaunama galutinė neurono klaida.

4. Judama per sluoksnius nuo pirmo iki paskutinio ir kiekvieno neurono svoriai ir slenkstis yra priderinami taip, kad neurono daroma klaida sumažėtų, tuo pačiu sumažinant viso tinklo klaidą. Neuronui paskaičiuojamos naujos jo svorių ir slenksčio reikšmės pagal šią formulę: $u = l \cdot (m \cdot u_0 + (1 - m) \cdot e \cdot X)$. Paaiškinimas:

   - u – svorio ar slenksčio atnaujinimo reikšmė. Ji yra pridedama prie dabartinės svorio ar slenksčio reikšmės, taip gaunant galutinę apmokytą reikšmę.
   - u_0 – svorio ar slenksčio atnaujinimo reikšmė gauta per prieš tai buvusį mokymą. Nevertinga jei impulsas (m) yra 0.
   - l – mokymosi koeficientas (angl. learning rate). Kontroliuoja svorių ir slenksčio atnaujinimo žingsnį. Reikšmė priklauso intervalui [0; 1]. Didelė šio kintamojo reikšmė leis DNT greičiau mokytis, tačiau su didesne tikimybe prašokti tinkamiausią svorio ar slenksčio reikšmę. Ir atvirkščiai, maža reikšmė privers mokytis lėtai, tačiau stabiliai.
   - m – impulsas (angl. momentum). Leidžia DNT greičiau mokytis įgyjant pagreitį judėti viena kryptimi link tinkamiausios svorio ar slenksčio reikšmės, taip išvengiant šokinėjimų į šonus. Reikšmė priklauso intervalui [0; 1].
   - e – trečiame žingsnyje apskaičiuota neurono klaida (angl. error).
   - X – reikšmė priklausanti nuo to ar tai svoris ar slenkstis:
     - Jei skaičiuojame atnaujinimą neurono slenksčiui: $X = 1$.
     - Jei skaičiuojame atnaujinimą neurono įvesties svoriui: X = įvesties, kuri jungiasi per šį svorį, reikšmei.

5. Kartojami 2-4 žingsniai kiekvienam įvesties-išvesties šablonui turimame mokymo duomenų rinkinyje.

6. Kartojamas 5 žingsnis pasirinktą skaičių iteracijų arba tol, kol pasiekiama pakankamai maža tinklo klaida. Vienas toks viso mokymo duomenų rinkinio praleidimas per tinklą yra vadinamas mokymosi epocha.

2. Programos kūrimas

Vienas iš pagrindinių šio darbo tikslų buvo parašyti programą, kuri sugebėtų apmokyti DNT iš paveiksliuko atpažinti, koks skaitmuo jame yra pavaizduotas. Prieš kuriant programą, jai buvo iškelti tokie pagrindiniai reikalavimai:

- *Duomenų paėmimas*. Rašytų skaitmenų paveiksliukai ir juos atitinkantys skaitmenys DNT apmokymui ir testavimui turi būti nuskaitomi iš failų.
- *Duomenų apdorojimas*. Iš nuskaitytų paveiksliukų turi būti ištraukiamos kokios nors jų savybės skaitmeniniu pavidalu (t. y. skaičių seka), kurios bus pateikiamos kaip įvestis į DNT.
- *Tinklo apmokymas*. DNT turi išmokti atpažinti skaitmenis iš pateikiamų paveikslėlių savybių. Mokymosi eiga turi būti pateikiama vartotojui išvedant į ekraną DNT klaidos rodiklius.
- *Tinklo išbandymas*. Programa turi leisti išbandyti apmokytą DNT su jam nematytais skaitmenų paveiksliukais. Turi būti pateikiamas DNT sėkmingų atpažinimų procentas. Taip pat vartotojas turėtų galėti grafiškai peržiūrėti bet kurį paveiksliuką, jo savybes (įvestį į DNT), jo vaizduojamą skaitmenį (išvestį iš DNT, kurios tikimasi) ir DNT rezultatą (tikrąją išvestį iš DNT).

Visi šie reikalavimai buvo patenkinti. Be to, programos kūrimo eigoje buvo pridėta ir papildomo funkcionalumo:

- Apmokyto DNT išsaugojimas ir užkrovimas iš .nnet failų (santrumpą paimiau iš angliškų žodžių „Neural NETwork“).
- Paveikslėlių apdorojimas: skeletizavimas ir histogramų radimas.
- Peržiūrimų paveiksliukų filtravimas pagal skaitmenis ir pagal tai, ar DNT atpažino skaitmenį.
- Ne tik originalių, bet ir modifikuotų (atlikus skeletizavimą) paveiksliukų peržiūra.

2.1. Programavimo įrankiai

2.1.1. C#

Programos kūrimui buvo naudojama nemokama programavimo aplinka „Microsoft Visual

C# 2008 Express Edition“. Ją galima nemokamai parsisiųsti iš Microsoft svetainės (http://www.microsoft.com/express/vcsharp/). Programa buvo rašoma C# kalba naudojant „.NET Framework” technologiją. Programai paleisti yra būtina tokia aplinka: Microsoft „Windows“ operacinė sistema su suinstaliuotu „.NET Framework 3.5”. Pasirinkau dirbti būtent su šiais įrankiais ir technologijomis, nes turiu daug patirties dirbant su jais. Be to, norėjau išbandyti vieną biblioteką (ši yra aprašyta toliau), kuri yra taip pat sukurta ant „.NET Framework“ platformos.

2.1.2. „AForge.NET“

Programos rašymui paspartinti naudojau atviro kodo biblioteką „AForge.NET“ (http://code.google.com/p/aforge/). Kad galėčiau stebėti ir analizuoti, kaip veikia šios bibliotekos įrankiai, parsisiunčiau jos kodą, susikompiliavau ir atlikau programos derinimą (angl. debug). Tai nebuvo sunku atlikti, kadangi ši biblioteka parašyta ta pačia programavimo kalba kaip ir mano programa – C#. Pasinaudojau bibliotekoje esančiais paveiksliukų apdorojimo („AForge.Imaging“) ir neuroninių tinklų („AForge.Neuro“) įrankiais. Iš paveiksliukų apdorojimo įrankių naudojau binarizavimo ir skeletizavimo algoritmus, kurių dėka buvo apdorojami paveiksliukai prieš atliekant savybių ištraukimą (angl. feature extraction) iš jų. Neuroninių tinklų įrankiai šioje bibliotekoje yra ganėtinai primityvūs, tačiau atlieka visas standartines DNT funkcijas, kurių man ir užteko savo darbo tikslams įgyvendinti. Tarp šių standartinių funkcijų galiu paminėti kelias, kurias pritaikiau savo programai. Pirmas pavyzdys – tinklo konstravimas iš neriboto kiekio neuronų sluoksnių, jungiant kiekvieno sluoksnio kiekvieną neuroną su kiekvienu neuronu iš kito sluoksnio. Kiti pavyzdžiai: tinklo apmokymas klaidos sklidimo atgal algoritmu ir dvipolė sigmoidinė aktyvavimo funkcija.

2.1.3. „Google Code“

Programos kūrimo metu buvo daug eksperimentuojama su DNT struktūra ir parametrais, išbandomi skirtingi problemų sprendimo būdai, kurie reikalavo dažno kodo keitimo. Taip dažnai viską keičiant pasitaikydavo nukrypimų į šoną, kas padarydavo programos veikimą neefektyvų ar mažiau efektyvų, nei šis buvo iki tol. Kad atstatyti programą į stabilią būseną reikėjo sekti jos kūrimo progresą. Čia prasivertė versijų kontrolės sistema „Subversion“ (SVN) ir „Google Code“ – svetainė, skirta atviro kodo projektų kūrimui. Šioje svetainėje susikūriau projektą savo programai (http://code.google.com/p/neural-net-digit-recognition/), o kodo failų perdavimui, gavimui, palyginimui ir kitai priežiūrai naudojau programą „TortoiseSVN“ (http://tortoisesvn.net/).

2.2. Duomenys – paveiksliukai ir jų vaizduojami skaitmenys

DNT apmokymui reikalingi labai dideli kiekiai skirtingų duomenų pavyzdžių, iš kurių mokydamasis DNT susidėliotų savo svorius taip, kad tinklas būtų tinkamas kuo platesnei įvairovei nematytų duomenų pavyzdžių.

Sprendžiamam uždaviniui reikėjo daug paveiksliukų su rašytais skaitmenimis. DNT apmokymams ir testavimui buvo naudota MNIST rašytų skaitmenų duomenų bazė, nes ji yra labai didelė ir tvarkinga. Taip pat, darbo vadovas ją rekomendavo. Ši duomenų bazė suskirstyta į du rinkinius. Pirmasis rinkinys yra skirtas DNT apmokymams. Jame yra 60000 paveiksliukų ir jų vaizduojamų skaitmenų. Antrasis rinkinys panašus į pirmąjį, tik šis yra skirtas DNT testavimui ir jame yra 10000 duomenų pavyzdžių. Kiekvienas rinkinys turi po du failus: viename saugomi paveiksliukai, kitame – skaičiai, nurodantys, kokius skaitmenis atitinkami paveiksliukai vaizduoja. Iš viso – keturi failai. Rinkiniai skiriasi tik dydžiais ir tuo, kad abiejuose yra skirtingų autorių rašyti skaitmenys. Mokomajame rinkinyje yra apie 250-ies skirtingų rašytojų skaitmenys, o testavimo rinkinyje rašytojų kiekis neaiškus, tačiau, kaip rašoma MNIST puslapyje [LC98], buvo pasirūpinta, kad rašytojų rinkiniai nesusikirstų duomenų rinkiniuose. Ši rinkinių savybė užtikrina testavimo tikslingumą, nes DNT turės atpažinti skaitmenis autorių, kurių rašto niekada nesimokė. Tai yra geras išbandymas neuroninio tinklo gebėjimui apibendrinti – iš apmokymo duomenų ištraukti tik esmines savybes, kurios yra būdingos ir duomenims, kurių jis niekada nematė. Mano sukurtas DNT šį išbandymą gana neblogai išlaikė. Skaitmenis, kuriuos mokėsi, atpažįsta 92,125 procentų tikslumu, o skaitmenis, kurių nesimokė atpažįsta 91,19 procentų tikslumu. Iš šių skaičių matome, kad nedidelis prisirišimas prie mokymosi duomenų yra.

Pagal MNIST puslapyje pateiktą instrukciją buvo parašytas modulis, kuris nuskaito iš skaitmenų duomenų bazių failų paveikslėlius ir juos atitinkančius skaitmenis.

2.3. Programos architektūra

Šiame skyriuje pateikiami galutinės programos struktūra ir veikimo principai, o kituose skyriuose daugiau dėmesio skiriama programos kūrimo eigai, spręstoms problemoms ir priimtiems sprendimams.

Programoje naudojamas DNT yra daugiasluoksnis perceptronas, kuris yra apmokomas klaidos sklidimo atgal algoritmu. Šis tinklas turi 3 sluoksnius: įvesties, vieną paslėptą ir išvesties.

DNT įvesties sluoksnis priima 166 įvesties reikšmes. Kadangi paveiksliukai yra 28x28 pikselių dydžio, tai vertikalios ir horizontalios histogramos turi po 28 reikšmes, o įstrižos

histogramos po 55 (žiūrėti priedo Nr.1 3 punktą). Visos keturių histogramų reikšmės sujungiamos į vieną 166 elementų masyvą (28+28+55+55=166), kuris ir yra pateikiamas į DNT.

Paslėptas sluoksnis turi 15 neuronų. Skaičius parinktas eksperimentavimo būdu, nes griežtos logikos, kaip parinkti vidinių DNT sluoksnių kiekį ir neuronų skaičių jame, nėra. Žinoma tik tiek, kad per mažas neuronų skaičius gali nesugebėti sumodeliuoti sprendžiamos problemos, o per didelis neuronų skaičius gali per daug tiksliai sumodeliuoti problemą būtent mokymosi duomenims ir prarasti gebėjimą apibendrinti, taip tapdamas netinkamas testavimo duomenims [Sta08].

Išvesties sluoksnis susideda iš 10 neuronų, o tai reiškia, kad DNT gražina 10 reikšmių. Kiekviena reikšmė nurodo tikimybę kiekvienam skaitmeniui. Kiekviena tikimybė kinta nuo -1 iki 1, kur didžiausių reikšmių pozicijos išvesties masyve parodo labiausiai tikėtiną skaitmenį. Pavyzdžiui, idealiu atveju, jei DNT išvestis yra masyvas, kurio nulinis elementas yra lygus 1, o kiti devyni elementai yra lygus -1, tai DNT yra įsitikinęs, kad atpažintas skaitmuo yra 0.

Programos logika suskirstyta į keturis etapus:

1. *Duomenų nuskaitymas*. Nuskaitomi paveiksliukai ir juos atitinkantys skaitmenys iš failų.
2. *Paveiksliuko apdorojimas* (angl. „preprocessing“). Taikomas skeletizavimo algoritmas, kad pašalinti storumo skirtumus tarp skirtingų paveiksliukų.
3. *Savybių ištraukimas*. Skaičiuojamos paveiksliuko histogramos.
4. *DNT apmokymas arba testavimas*. Histogramos pateikiamos kaip įvestis į DNT ir gaunama išvestis.
    1. Jei vykdomas tinklo apmokymas, tai mokytojas (mokymo algoritmas) palygina gautą išvestį su laukta išvestimi ir pagal gautą paklaidą tarp šių išvesčių apmoko DNT (t. y. pakeičia DNT svorius) klaidos sklidimo atgal algoritmu, kaip aprašyta 1.2.1. skyriuje. Kiekviena įvedimo-išvedimo paklaida yra sumuojama ir atlikus apmokymus su visais duomenimis yra pateikiama kaip galutinė mokymosi klaida (žiūrėti priedo Nr.2 mokymosi rezultatus).
    2. Jei vykdomas tinklo testavimas, tai iš išvesties reikšmių paimama didžiausia reikšmė. Jei ji priklauso skaitmeniui, kurio buvo tikimasi, tai teisingų atspėjimų skaitliukas padidinamas vienetu. Atlikus testavimą su visais duomenimis, galutinė skaitliuko reikšmė padalinama iš duomenų kiekio ir gaunamas teisingai atpažintų skaitmenų procentas. (žiūrėti priedo Nr.2 testavimo rezultatus).

2.4. Paveiksliukų apdorojimas ir savybių ištraukimas

Sunkiausias uždavinys buvo tinkamų savybių parinkimas. Po ilgų tyrinėjimų ir straipsnių skaitymų buvo apsispręsta pasirinkti naudoti histogramas kaip paveiksliukų savybes ir įvestį į DNT. Šis apsisprendimas daugiausiai buvo įtakotas vieno skaityto darbo [Wu03], kuriame autorius tikino sėkmingai pritaikęs šį būdą. Iš paveiksliuko yra paskaičiuojamos keturios histogramos: horizontali, vertikali, įstriža į kairę ir įstriža į dešinę. Histograma yra gaunama paimant paveiksliuko pikselių eilutę, stulpelį arba įstrižainę (priklausomai nuo to, kokios histogramos ieškome) ir sudedant visas jos pikselių reikšmes, taip gaunant vieną reikšmę. Tas pats daroma su visomis eilutėmis, stulpeliais ar įstrižainėmis, taip gaunant masyvą reikšmių. Po to visi keturi masyvai (kiekvienai histogramai) sujungiami į vieną masyvą, kuris yra naudojamas kaip įvestis į DNT.

2.4.1. Histogramų reikšmių optimizavimas

Su tokiomis pradinėmis histogramomis tinklas veikė labai nesėkmingai, nes histogramų reikšmės buvo labai didelės. Šios kartais siekdavo 5000 ir daugiau, kai vienoje tiesėje būdavo po apie 20 baltų pikselių (255 * 20 = 5100, čia 255 yra balto pikselio reikšmė). Tinklo neuronų svoriai inicializuojami atsitiktinėmis reikšmėmis iš intervalo [0; 1], o neuronas skaičiuoja išvestį sudėdamas visas įvesties reikšmes sudaugintas su atitinkamais svoriais ir perleisdamas šią sumą, kuri siekdavo šimtus tūkstančių, per dvipolę sigmoidinę aktyvavimo funkciją. Rezultatas beveik visada gaudavosi vienetas, nes naudojama funkcija generuoja rezultatus iš intervalo [-1; 1], o jautriausia ji yra įvestims iš intervalo [-e; e], kur e yra matematinė konstanta ≈ 2,71828. Akivaizdžiai, keli tūkstančiai netelpa į šį intervalą. Nors apmokymo metu svoriai vis tik mažėdavo, tad ir suma atitinkamai mažėdavo, tačiau tai vyko per lėtai ir per 15 apmokymo epochų (viena epocha – 60000 apmokymų skirtingomis įvestimis) DNT nieko neišmokdavo, t. y. atpažinimo procentas likdavo panašus kaip ir prieš apmokymą – apie dešimt procentų. Ši problema buvo sprendžiama šiek tiek pakeičiant histogramų reikšmes. Pirmu bandymu jos buvo tiesiog suspaustos į intervalą [0; 1]. Po šio pakeitimo tinklas pradėjo žymiai greičiau konverguoti ir jau po kelių apmokymo epochų pradėdavo gražinti ne tik vienetus. Po apmokymų buvo pasiektas virš 80 procentų DNT tikslumas. Buvo galima matyti, kad per pirmas kelias epochas tinklas vistiek dar turėdavo tą pačią problemą – neuronai bet kokioms įvestims generuodavo tą pačią išvestį – vienetą. Sprendžiant šią problemą tapo akivaizdu, kad reikalingos ne tik teigiamos pradinių svorių ir įvesčių reikšmės, bet ir neigiamos. Čia pakeitimas nebuvo sudėtingas – buvo padaryta, kad histogramų reikšmių suma būtų lygi nuliui, t. y. iš kiekvienos reikšmės atimamas

visų reikšmių vidurkis. Reikšmės po pakeitimo pradėjo svyruoti intervale [-1; 1] (žiūrėti priedo Nr.1 5 ir 6 punktus). Kaip ir buvo tikėtasi, DNT pradėjo jau pirmos epochos mokymosi metu daryti žymiai mažesnes klaidas, t. y. skirtumai tarp gautų išvesties reikšmių ir reikšmių, kurių buvo tikimasi, tapo gana maži.

2.4.2. Paveiksliukų filtravimas

Paveiksliukuose labai skiriasi rašytų skaitmenų linijų storiai, o tai labai įtakoja histogramas. Kuo linijos storesnės, tuo labiau kai kurie histogramų stulpeliai iššoka virš kitų. Dėl to, kai kurie rašyti skaitmenys, nors ir vizualiai būdami labai panašūs, turi gana skirtingas histogramas. Kad pašalinti šią problemą, paveiksliukams buvo pritaikytas skeletizavimas. Dėka šio algoritmo paveiksliukų histogramų skirtumai sumažėjo. Kaip pasekmė, DNT atpažinimo procentas išaugo pora procentų (nuo maždaug 88 iki 90). Buvo tikėtasi didesnio pagerėjimo, tačiau neuronų tinklas parodė, kad nedidelė įvesties įvairovė neturi didelės įtakos jo atsakymui. Taigi neuronų tinklas moka pritaikyti net gana skirtingas histogramas tiems patiems skaitmenims atpažinti.

3. Palyginimas su kitų autorių darbais

Šiame skyriuje pateikiamas mano rašytų skaitmenų atpažinimo problemos sprendimo palyginimas su poros kitų autorių sprendimais.

Miguel Po-Hsien Wu, darbo „Handwritten Character Recognition“ [Wu03] autorius, rašo, kad sukūrė programą, kuri naudoja trijų sluoksnių DNT, apmokomą klaidos sklidimo atgal mokymo algoritmu. Kadangi darbo metu daugiausiai nagrinėjau šio tipo DNT, tai mano programa taip pat naudoja panašios struktūros DNT. Autorius savo programoje naudoja paveikslėlių histogramas kaip įvestį į DNT ir tvirtina pasiekęs 99 procentų skaitmenų atpažinimo tikslumą. Kadangi dauguma sudėtingesnių metodų (pvz.: konvoliuciniai (angl. „convolutional“) DNT, atramos vektorių mašinos (angl. „support vector machines“)) vos pasiekia tokį gerą rezultatą, buvo pasirinkta išnagrinėti šio rezultato priežastį ir parodyti tikrąjį šio metodo gebėjimą klasifikuoti. Mano programa, taikydama histogramas įvesčiai į DNT, pasiekė apie 91 procento tikslumą atpažįstant rašytinius skaitmenis. Pagrindinė priežastis, kodėl šis rezultatas atsiliko nuo Wu pasiekto rezultato, yra tai, kad mano programa teisingai naudojo apmokymo ir testavimo duomenis. Autorius savo darbe rašo, kad naudojo po šimtą kiekvieno skaitmens pavyzdžių. Iš viso tai yra tūkstantis paveiksliukų, kuriuos autoriaus programa naudojo tiek DNT apmokymams, tiek ir testavimui. Mano programa naudojo 60000 skaitmenų pavyzdžių tinklo apmokymams ir 10000 – tinklo testavimui iš MNIST duomenų bazės. Šis testavimo ir mokymosi duomenų atskyrimas ir lėmė prastesnį, tačiau korektiškesni, rezultatą. Palyginus su Wu, mano DNT mokėsi iš žymiai daugiau duomenų ir konstravo labiau apibendrintą sprendimą, taip mažiau prisirišdamas prie mokymosi duomenų. Kadangi nėra aišku, kaip autoriaus programai sektųsi atpažinti skaitmenis, kurių ji niekada nematė, tai sunku palyginti jos efektyvumą su mano programos efektyvumu.

Labai gera rašytinių skaitmenų atpažinimo programa ir jos detalus aprašymas [One06] yra pateikiami Mike O'Neill. Šis autorius rėmėsi Dr. Yann LeCun [LBB+98] ir Dr. Patrice Simard konvoliucinių neuroninių tinklų ir vaizdinių dokumentų atpažinimo darbais. Programos architektūra, kaip teigia autorius, buvo pagrįsta būtent šiuose darbuose aprašytais metodais. O'Neill savo programoje naudoja penkių sluoksnių konvoliucinį neuroninį tinklą (KNT). Šis tinklas atpažįsta vaizdinius šablonus tiesiai iš paveiksliukų pikselių su minimaliu išankstiniu paveiksliukų apdorojimu. KNT gali atpažinti šablonus su labai dideliais skirtumais (tokius kaip ranka rašytus simbolius). Taip pat jie yra labai atsparūs paveikslėlių iškraipymams ir geometrinėms transformacijoms. Dėl šių priežasčių O'Neill programa pasiekė 99,26 procentų

tikslumą atpažįstant skaitmenis iš MNIST duomenų bazės. Kadangi mano sukurto DNT apmokymai ir testavimai buvo vykdomi irgi ant MNIST duomenų, tai akivaizdžiai matosi KNT pranašumas lyginant su DSP ir jo įvesčiai naudojamomis histogramomis. KNT modeliuoja sudėtingesnę problemą, kadangi turi žymiai daugiau įvesčių (kiekvienas pikselis yra įvestis) ir žymiai daugiau vidinių sluoksnių (net tris, palyginti su mano DNT naudotu vienu). Mano DNT dirba ne su pilna paveiksliuko informacija, nes histogramose neatsispindi visi paveiksliuko požymiai. Kai kurios histogramos skirtingiems vaizduojamiems skaitmenims tiesiog per daug panašios (pvz.: skaitmenų '4' ir '9' paveiksliukai, kurie tinklui ir buvo sunkiausiai atskiriami – sudarė net apie 14% visų neatpažinimų), o tiems patiems vaizduojamiems skaitmenims per daug skiriasi dėl skaitmens iškraipymo ar geometrinės transformacijos, kurioms histogramos yra gana jautrios. Šios transformacijos KNT didelio poveikio neturi. Teigiamų dalykų yra ir mano naudotame metode. Vienas iš pagrindinių yra mokymosi greitis. O'Neill savo straipsnyje rašo, kad jo tinklo apmokymas trukdavo apie 14-16 valandų. Mano naudojamam tinklui apmokyti užtenka vos 15 minučių. Čia reikėtų paminėti, kad mano naudoto kompiuterio parametrai buvo netgi prastesni už autoriaus. Autoriaus DNT apmokyti (tinklas laikomas apmokytu tada, kai jo daroma klaida beveik nebemažėja) taip pat prireikdavo daugiau mokymo epochų (20-25) negu mano DNT (15 epochų). Pagrindinė to priežastis yra tinklo dydis – kuo tinklas didesnis, tuo daugiau jį reikia mokyti.

Išvados ir rekomendacijos

Dirbtiniai neuroniniai tinklai (DNT) yra puikiai pritaikomi sudėtingų uždavinių sprendimuose. Šie uždaviniai dažniausiai neturi algoritminio, loginio ir tiesiogiai prieinamo sprendimo būdo arba jeigu turi, tai sprendimas nėra tinkamos kokybės. Vienas iš tokių uždavinių pavyzdžių yra rašytų skaitmenų atpažinimas. Tai yra klasifikavimo uždavinys, kurį, kaip buvo parodyta darbe, gana sėkmingai sprendžia DNT. Pagrindinė šios sėkmės priežastis yra DNT gebėjimas sumodeliuoti apibendrintą problemos sprendimą mokantis iš pateikiamų duomenų. DNT taip pat sugeba dirbti su triukšmingais ir labai įvairiais duomenimis. Skirtingų autorių rašyti skaitmenys būtent tokie ir būna.

Darbe buvo išnagrinėti DNT, jų architektūra ir veikimo principai. Buvo išanalizuoti daugiasluoksniai perceptronai – vieni iš populiariausių neuroninių tinklų. Taip pat buvo apžvelgti DNT mokymosi principai ir giliau išanalizuotas klaidos sklidimo atgal mokymosi algoritmas.

Buvo atliktas praktinis DNT teorijos pritaikymas – sukurta programa, kuri atpažįsta rašytus skaitmenis. Ši programa naudojo anksčiau minėtus daugiasluoksnius perceptronus ir klaidos sklidimo atgal algoritmą. Įvesčiai į DNT buvo naudojamos skeletizuotam paveiksliukui apskaičiuotos histogramos. Pasiektas neblogas apie 91 procento skaitmenų atpažinimo tikslumas. Gautas rezultatas yra patikimas, nes DNT apmokymui ir testavimui buvo naudojama populiari MNIST rašytų skaitmenų duomenų bazė, kurioje yra dešimtys tūkstančių rašytų skaitmenų pavyzdžių. Programoje realizuotas sprendimo metodas buvo palygintas su kitų autorių darbuose aprašytais metodais. Iš šių palyginimų buvo galima matyti, kad mano pasirinktas sprendimo būdas nėra pats tinkamiausias šiam klasifikavimo uždaviniui spręsti, nes histogramos nepilnai apibrėžia, koks skaitmuo yra vaizduojamas paveiksliuke. Kai kuriems skirtingų skaitmenų paveiksliukams histogramos yra labai panašios, o kai kuriems tų pačių skaitmenų paveiksliukams histogramos yra labai skirtingos.

Parašytos programos tobulinimui reikėtų keisti naudojamą DNT architektūrą ir pritaikyti konvoliucinius neuroninius tinklus ar kokius kitus sprendimo būdus, kurie sugebėtų tiksliau modeliuoti įvesties-išvesties šablonus.

Literatūros sąrašas

[AM92] Dave Anderson, George McNeill. Artificial neural networks technology. Kaman Sciences Corporation, 1992, 87 pages.

[DHS01] Richard O. Duda, Peter E. Hart, David G. Stork. Pattern Classification. Wiley-Interscience, 2001, p.282-299.

[KS96] Ben Krose, Patrick van der Smagt. An introduction to Neural Networks. The University of Amsterdam. 1996. p.13-37.

[LBB+98] Y. LeCun, L. Bottou, Y. Bengio, P. Haffner. Gradient-Based Learning Applied to Document Recognition. Proceedings of the IEEE, 86(11):2278-2324, November 1998, 46 pages.
URL: http://yann.lecun.com/exdb/publis/pdf/lecun-98.pdf.

[LC98] Yann LeCun, Corinna Cortes . The MNIST database of handwritten digits. 1998.
URL: http://yann.lecun.com/exdb/mnist/.

[Mak06] Rūta Makūnaitė. Neuroniniai tinklai. Žurnalas „Kompiuterija“, 2006, p.37-38.

[One06] Mike O'Neill. Neural Network for Recognition of Handwritten Digits. CodeProject, 2006.
URL: http://www.codeproject.com/KB/library/NeuralNetRecognition.aspx.

[Sta08] StatSoft. Neural Networks. StatSoft, Inc., 2008.
URL: http://www.statsoft.com/textbook/stneunet.html.

[Wu03] Miguel Po-Hsien Wu. Handwritten Character Recognition, The University of Queensland, 2003.
URL: http://innovexpo.itee.uq.edu.au/2003/exhibits/s804636/thesis.pdf. 456KB.

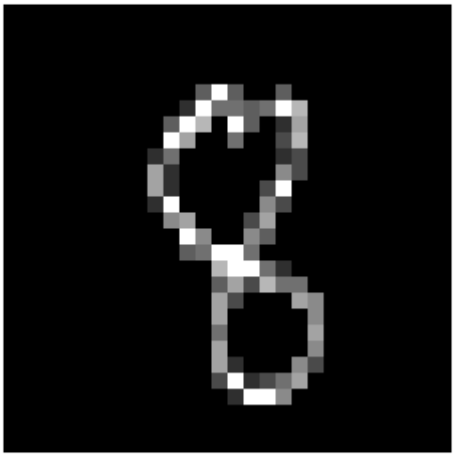
[Zil01] Ali Zilouchian. Intelligent Control Systems Using Soft Computing Methodologies. CRC Press LLC, 2001, p.17-38.

Priedas Nr.1 – Paveiksliukų savybių ištraukimas

1. Nuskaitomas paveiksliukas iš failo. Paveiksliukas jau iš anksto apdorotas tokiomis operacijomis (kaip aprašyta MNIST puslapyje [LC98]):

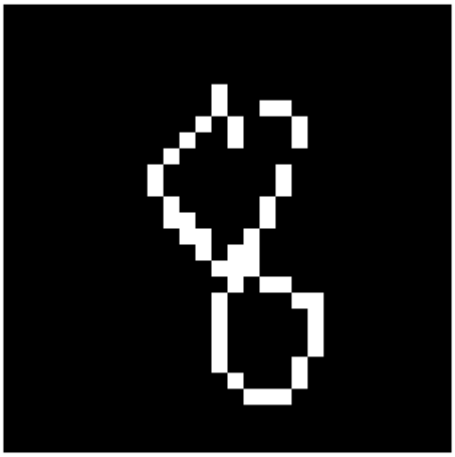
   I. Originalaus juodai balto (dviejų lygių) paveiksliuko iš NIST dydis buvo normalizuotas taip, kad paveiksliukas tilptų į 20x20 pikselių rėmą išlaikant jo aukščio ir pločio santykį. Gautas paveiksliukas turi pilkų lygių, kas yra rezultatas „anti-aliasing“ technikos naudotos normalizavimo algoritme.

   II. Toliau paveiksliukas buvo sucentruotas į 28x28 pikselių dydžio paveiksliuką paskaičiavus pikselių masės centrą ir pastūmus paveiksliuką taip, kad šis centras būtų per vidurį 28x28 dydžio pikselių lauko.



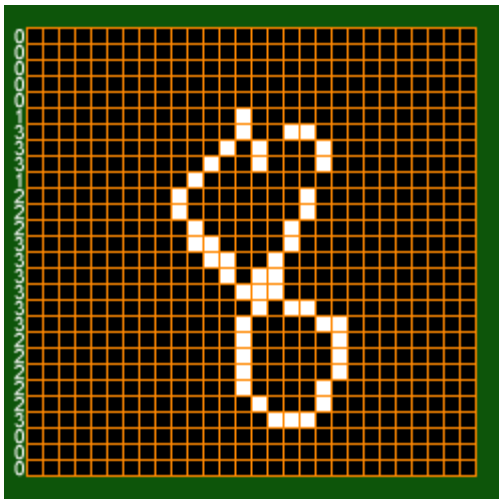
Pradinis paveiksliukas.

2. Atliekamas skeletizavimas.

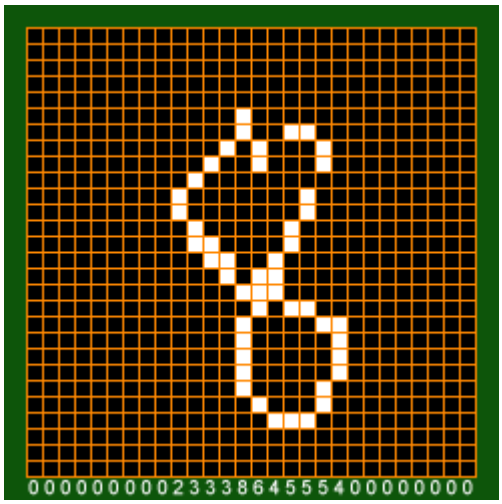


Skeletizuotas paveiksliukas.

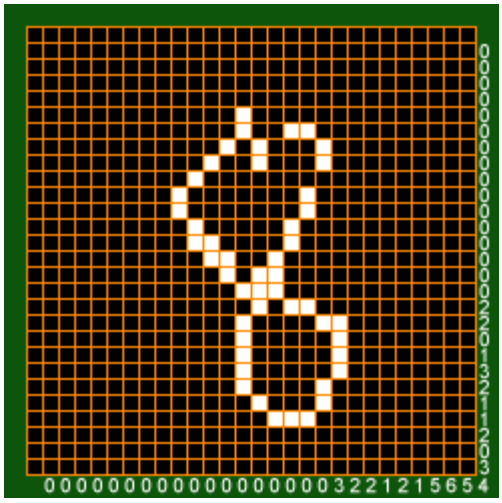
3. Apskaičiuojamos histogramos.



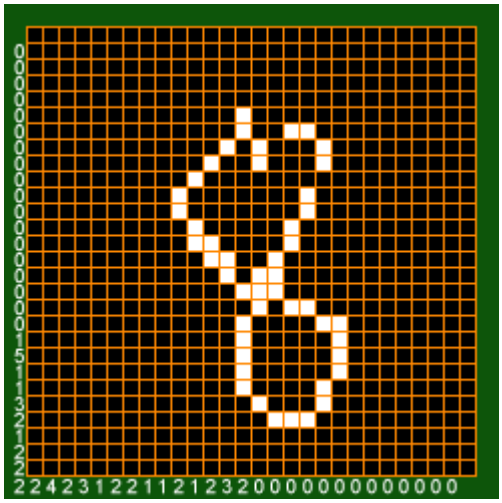
Horizontalios histogramos.



Vertikalios histogramos.



Kairinės įstrižos histogramos.



Dešininės įstrižos histogramos.

4. Histogramos sujungiamos į vieną skaičių masyvą:

   00000133312223333332222223000

   00000000023338645554000000000

   0000000000000000003221215654302112310220000000000000000000

   0000000000000000001511321222242312211212320000000000000000

5. Randamas masyvo didžiausias elementas ir paskaičiuojamas visų elementų vidurkis.

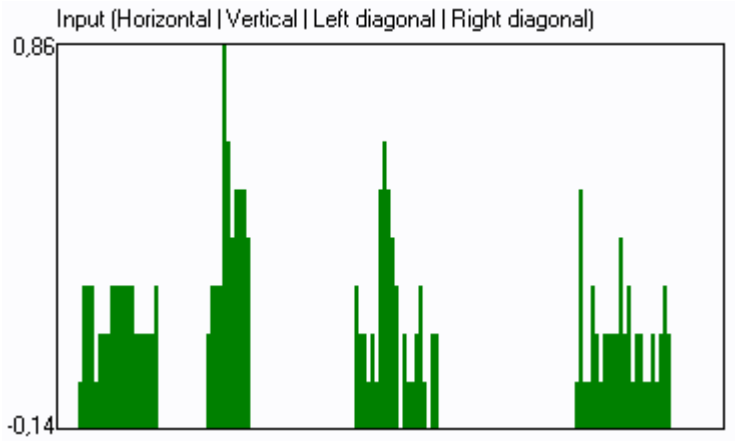
   Didžiausias elementas: 8

   Elementų vidurkis: 1,1566

6. Iš kiekvieno masyvo elemento atimamas elementų vidurkis ir tada padalinama iš didžiausio elemento. Taip užtikrinama, kad visų elementų suma bus lygi 0, ir visi elementai bus iš intervalo [-1; 1].

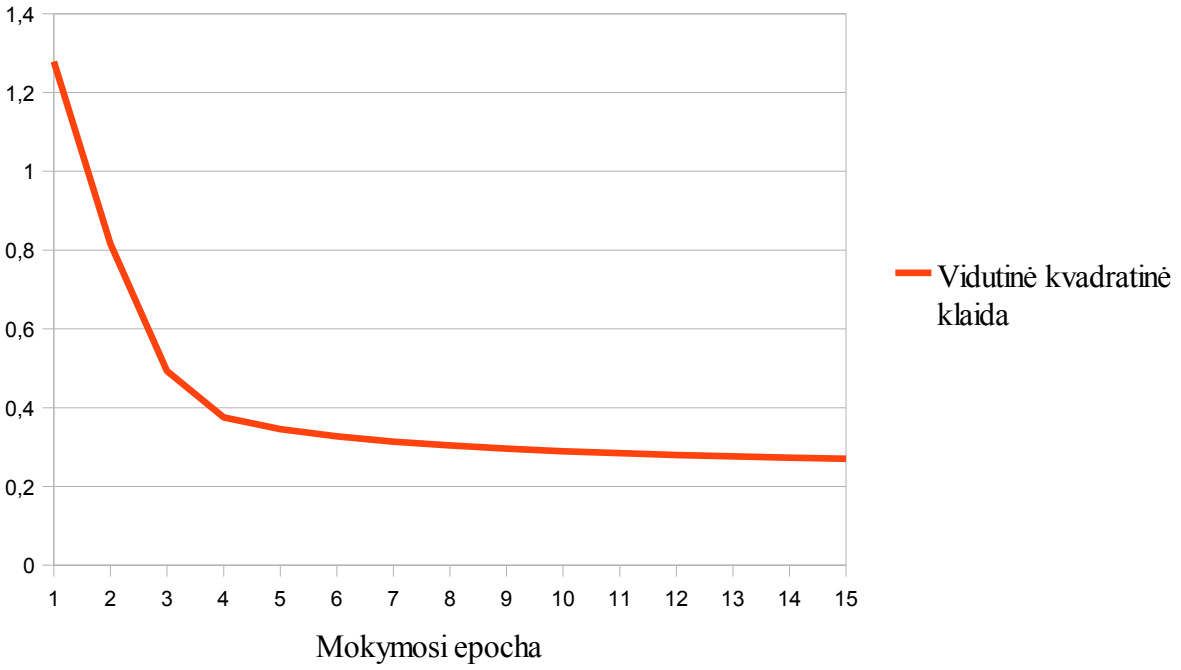
   Naujas mažiausias elementas: -0,1446

   Naujas didžiausias elementas: 0,8554

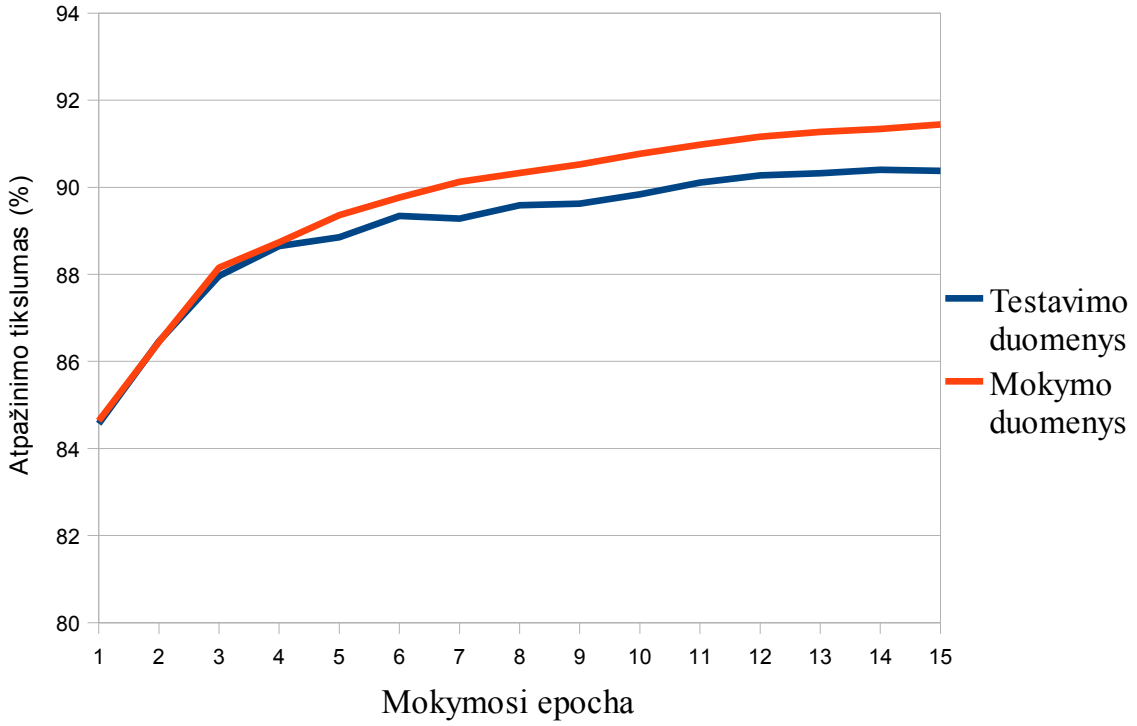


Paveiksliuko savybės (histogramos) atvaizduotos grafiškai. Šis reikšmių masyvas yra įvestis į DNT.

Priedas Nr.2 – Mokymosi ir testavimo rezultatai



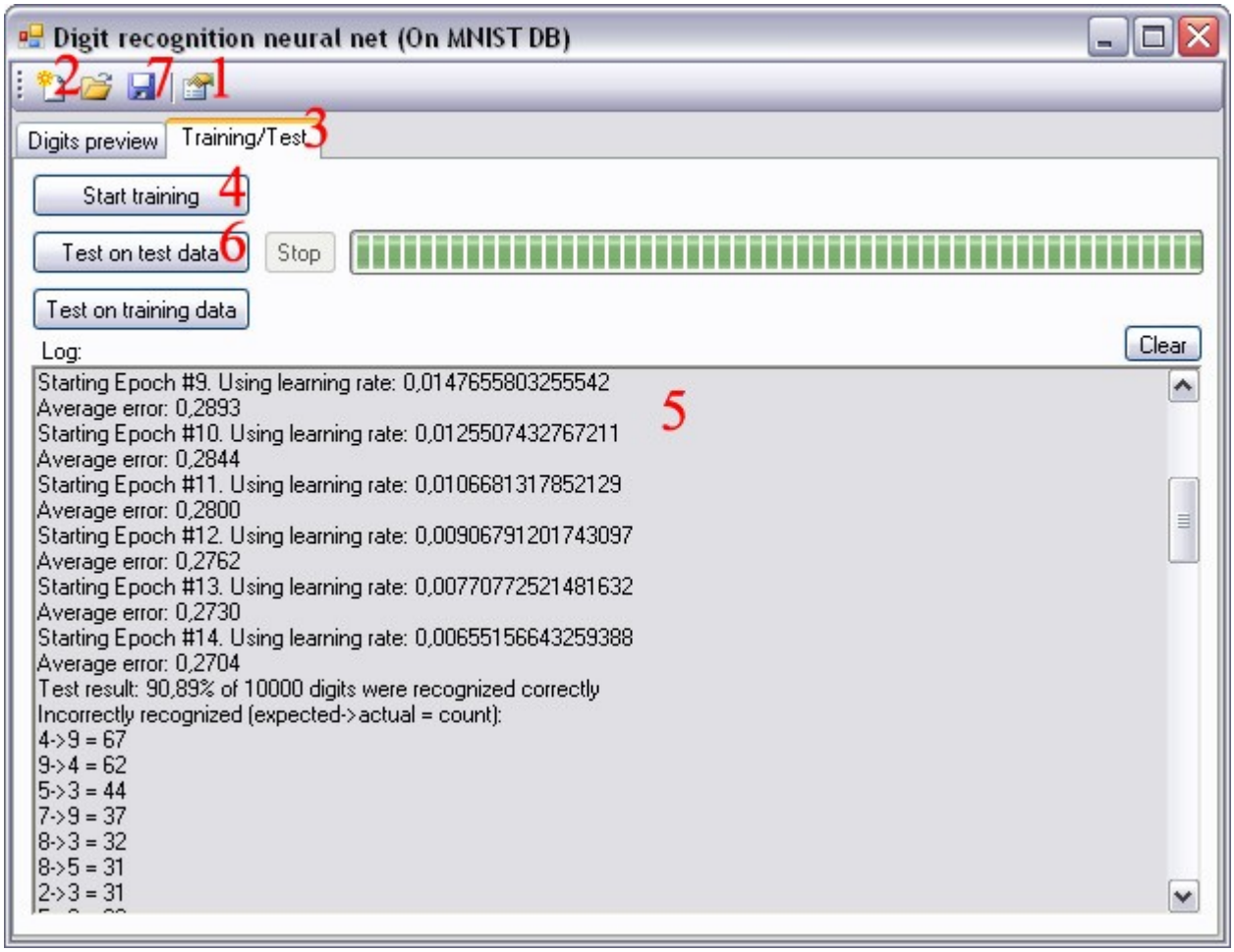
Galutinio DNT mokymosi rezultatai. Per pirmas kelias mokymosi epochas tinklas sparčiai mažina savo klaidas, o po to beveik nebetobulėja, nes jau įsisavino pagrindinius įvesties šablonus.



Galutinio DNT testavimo rezultatai. Mokomasi su mokymosi duomenimis, o testuojama ir su testavimo ir su tais pačiais mokymosi duomenimis po kiekvienos epochos. Matosi, kad jau po 4 epochų DNT pradeda prisirišti prie mokymosi duomenų ir gebėjimas apibendrinti po truputi mažėja.

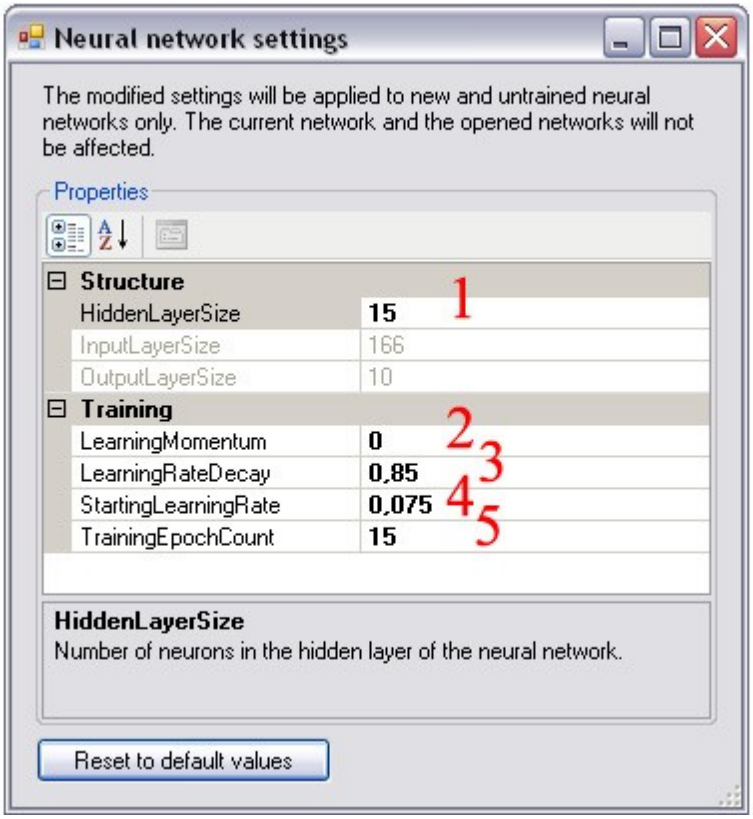
Priedas Nr.3 – Programos naudojimas

1. Dirbtinio neuroninio tinklo apmokymas ir testavimas



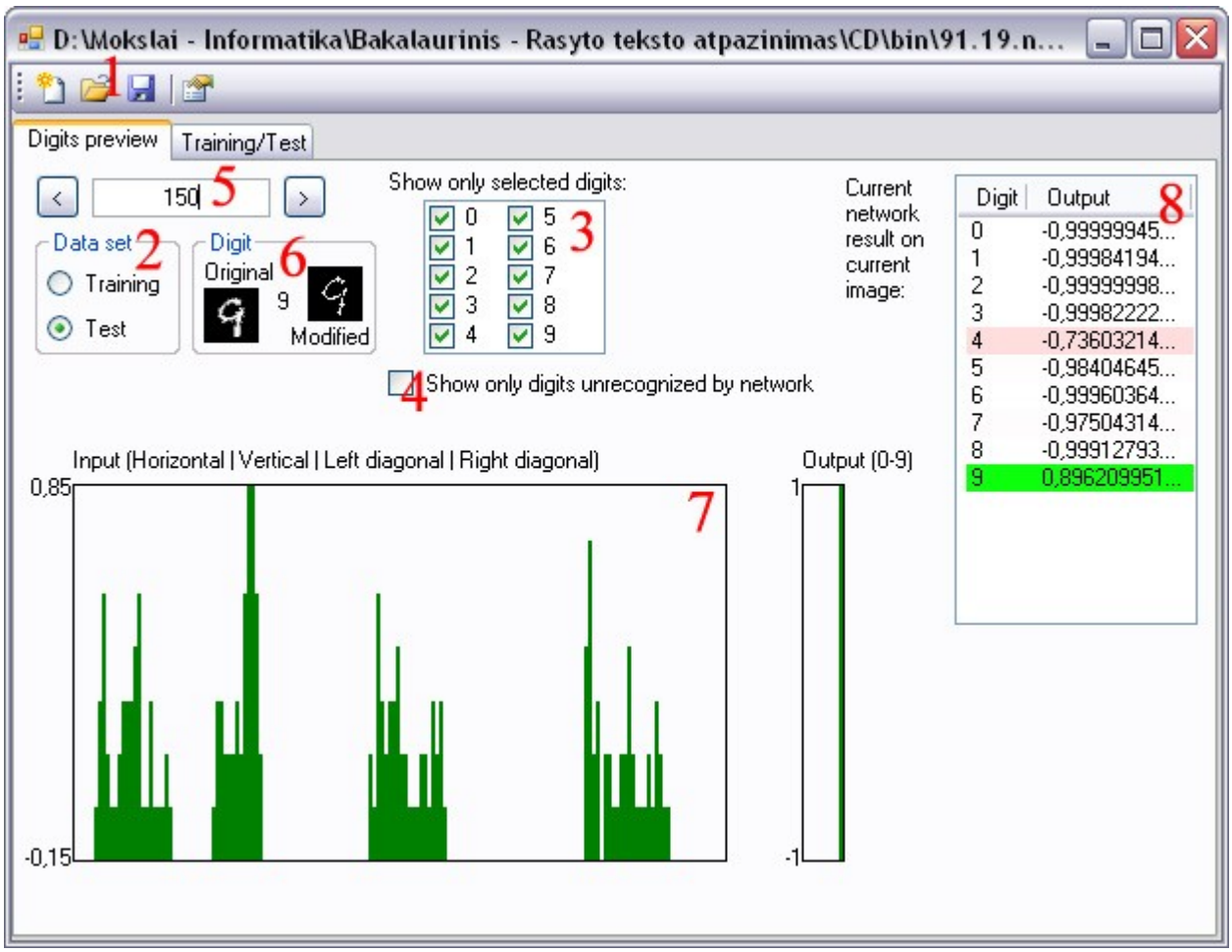
1. Atidaromas DNT parinkčių langas (žiūrėti šio priedo 2-ą skyrių), kuriame nustatomi DNT mokymosi greitis, epochų kiekis ir kiti parametrai.
2. Sukuriamas naujas neapmokytas DNT su 1-ame žingsnyje nustatytais parametrais.
3. Atverčiamas mokymo-testavimo puslapis.
4. Pradedamas mokymas. Kadangi šis procesas užtrunka šiek tiek laiko, tai yra rodomas progresas.
5. Atspausdinami mokymo rezultatai po kiekvienos mokymo iteracijos (epochos).
6. Pradedamas jau apmokyto DNT testavimas. Rezultatai taip pat atspausdinami apatiniame teksto lauke (5).
7. Išsaugomas DNT į pasirinktą failą, kad vėliau jį būtų galima užkrauti ir naudoti.

2. Dirbtinio neuroninio tinklo parametrų parinkimas



1. Parenkamas DNT paslėptojo sluoksnio neuronų kiekis. DNT įvesties ir išvesties reikšmių kiekio keisti negalima.
2. Parenkamas mokymosi impulsas (aprašytas 1.2.1. skyriuje).
3. Parenkamas mokymosi greičio nuvertėjimo koeficientas. Šis skaičius naudojamas po kiekvienos mokymosi epochos, kad sumažinti mokymosi greitį.
4. Parenkamas pradinis mokymosi greitis (aprašytas 1.2.1. skyriuje).
5. Parenkamas mokymosi iteracijų (epochų) kiekis. Šis skaičius nurodo, kiek kartų bus kartojamas mokymas su visais mokymosi duomenimis.

3. Duomenų peržiūra ir dirbtinio neuroninio tinklo išbandymas



1. Užkraunamas anksčiau išsaugotas DNT iš pasirinkto failo.
2. Pasirenkama, kurį duomenų rinkinį yra norima peržiūrėti.
3. Pasirenkama, kokius skaitmenis yra norima peržiūrėti.
4. Pasirenkama, ar yra norima matyti tik tuos skaitmenų paveiksliukus, kurių DNT neatpažino.
5. Pasirenkama, kelintą paveiksliuką ir jį atitinkantį skaitmenį yra norima peržiūrėti iš MNIST duomenų bazės failų. Mokymo duomenų yra 60000, testavimo – 10000.
6. Atvaizduojamas originalus paveiksliukas (iš failo), jį atitinkantis skaitmuo ir skeletizuotas paveiksliukas.
7. Pateikiamos sujungtos skeletizuoto paveiksliuko histogramos.
8. Pateikiamas DNT išvestis pasirinktam paveiksliukui. Skaitmuo, ties kuriuo yra didžiausia

išvesties reikšmė, yra DNT atpažintas skaitmuo. Teisingas skaitmuo nuspalvinamas žalia spalva, visi kiti – raudona. Spalvos intensyvumą nurodo išvesties reikšmė (-1 – nematoma, 0 – blyški, 1 – labai ryški).